Instituto Politécnico Nacional

Escuela Superior De Computo

Sistemas en Chip

Práctica 1

Nombre de los integrantes:

García Quiroz Gustavo Iván

Bejarano García Owen Uriel

Grupo: 7CV3

Nombre Del Profesor: Miguel Ángel Castillo Martínez

Fecha De Entrega: 25/03/2025

Índice

Introducción 1

Marco teórico 2

UART 2

Formato de trama en UART 2

Materiales y Equipo Utilizado 4

Desarrollo de la Práctica 5

Actividad 1: Frecuencia del Procesador (Onda Cuadrada) 5

Actividad 2: Transmisión UART 6

Conclusiones 12

Referencias 13

Introducción

En esta práctica, se exploraron dos aspectos clave de los sistemas en chip (SoC): la frecuencia de operación del microcontrolador y la comunicación serial asíncrona (UART).

El Arduino Uno, basado en el microcontrolador ATmega328P, opera a una frecuencia de 16 MHz, lo que permite ejecutar instrucciones en tiempos determinados. Mediante el osciloscopio, se analizó una señal de onda cuadrada generada por el Arduino para verificar su comportamiento en el dominio del tiempo.

Por otro lado, el protocolo UART es ampliamente utilizado en sistemas embebidos para transmisión de datos serie sin necesidad de señal de reloj. En esta práctica, se implementó un código en C para enviar un carácter ASCII (la letra 'A') a través del puerto serial, visualizando su estructura de bits en el osciloscopio. Esto permitió comprender la trama UART, compuesta por:

- Un bit de start (nivel bajo).
- 8 bits de datos (en orden LSB-first).
- Un bit de stop (nivel alto).

El objetivo principal fue validar experimentalmente el funcionamiento de estos conceptos teóricos, utilizando herramientas como el osciloscopio Rigol para observar señales digitales y confirmar su correspondencia con los valores esperados. Los resultados obtenidos sirven como base para aplicaciones más avanzadas, como la comunicación entre microcontroladores y sensores en sistemas IoT.

Marco teórico

UART

UART (universal asynchronous receiver / transmitter, por sus siglas en inglés) define un protocolo o un conjunto de normas para el intercambio de datos en serie entre dos dispositivos. UART es sumamente simple y utiliza solo dos hilos entre el transmisor y el receptor para transmitir y recibir en ambas direcciones. Ambos extremos tienen una conexión a masa. La comunicación en UART puede ser simplex(los datos se envían en una sola dirección), semidúplex(cada extremo se comunica, pero solo uno al mismo tiempo), o dúplex completo(ambos extremos pueden transmitir simultáneamente). En UART, los datos se transmiten en forma de tramas.

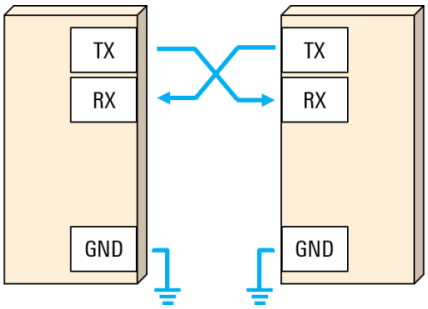


Figura 1 UART

Formato de trama en UART

Figura 2 Formato de trama en UART

Como en la mayoría de los sistemas digitales, un nivel de tensión alto se utiliza para indicar un 1 lógico, y un nivel de tensión bajo se emplea para indicar un 0 lógico. Dado que el protocolo UART no define tensiones específicas o rangos de tensión para estos niveles, a veces se denomina al nivel alto marca y al bajo espacio.

Obsérvese que en el estado de reposo (cuando no se transmiten datos) la línea se mantiene en el estado alto. Esto permite detectar con facilidad una línea o un transmisor averiado.

Materiales y Equipo Utilizado

Material/Equipo	Especificaciones
Arduino Uno	Microcontrolador ATmega328P, 16 MHz
Osciloscopio Rigol	Modelo DS1054Z (4 canales)
Cables de conexión	Jumpers MM y HM
Computadora	Con IDE Arduino instalado

Desarrollo de la Práctica

Actividad 1: Frecuencia del Procesador (Onda Cuadrada)

Para analizar la frecuencia de operación del Arduino, se implementó un programa básico que configura todos los pines del puerto D como salidas y luego intenta cambiar el estado de los pines del puerto B en un bucle infinito.

```
int main(){
  DDRD = 0xff;
  while(1){
    PINB = 0xff;
  }
}
```

Figura 3 Programa básico

Configuración del Osciloscopio

- Conexión: Sonda CH1 en pin 3 del Arduino.
- Escala de tiempo: 200 ns/div (para ver el periodo de la señal).
- Escala de voltaje: 20V/div (señal TTL de 0V a 20V).
- Trigger: Edge (flanco de subida).

Resultados Esperados

- Frecuencia medida:
    - Teórica: **16 MHz** (periodo = 62.5 ns).
    - Observada: Debido a limitaciones del digitalWrite(), la frecuencia será mucho menor (kHz), pero se puede ver la forma de onda cuadrada.

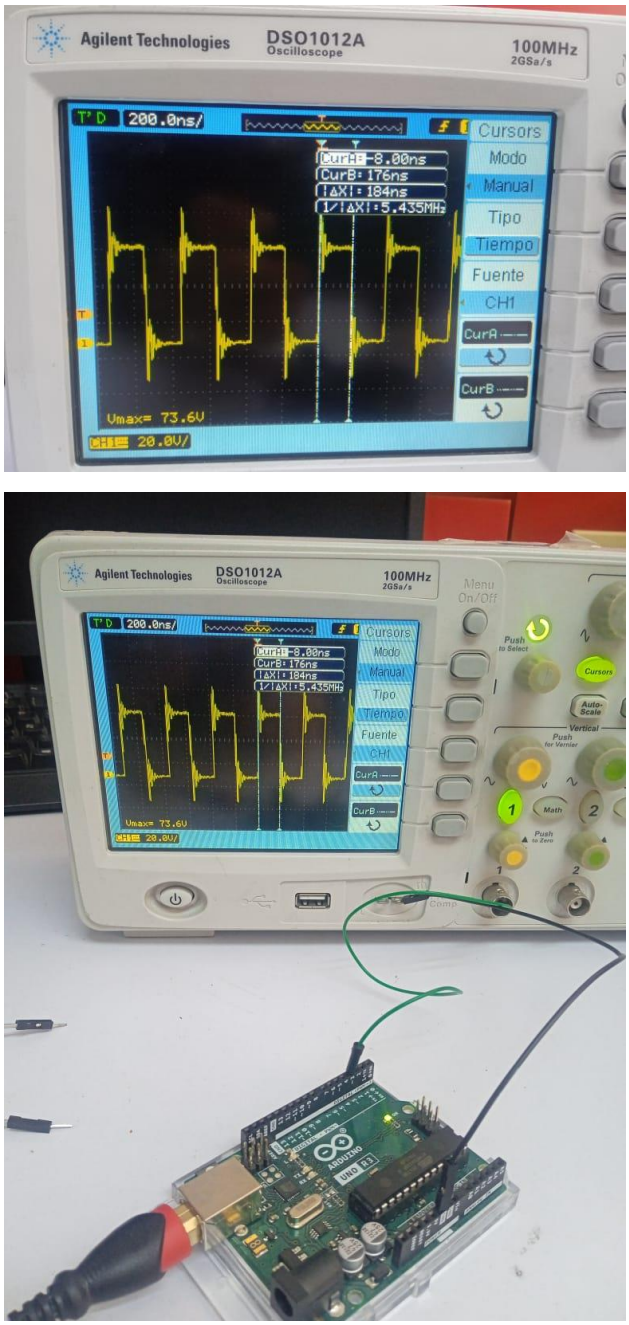


Figura 4

Actividad 2: Transmisión UART

En esta actividad se implementó un programa en C para configurar el módulo UART del ATmega328P y transmitir la letra 'A' (código ASCII 0x41) cuando se activaba una interrupción externa en el pin 2. El código inicializó el UART a 9600 baudios,

con 8 bits de datos y 1 bit de parada, utilizando los registros UBRR0, UCSR0B y UCSR0C del microcontrolador.

Al conectar el osciloscopio al pin TX (D1) del Arduino y activar la interrupción, se capturó la trama UART completa. La señal mostró claramente el bit de start (nivel bajo), seguido de los 8 bits de datos (10000010, enviados LSB primero) y finalmente el bit de stop (nivel alto).

```c
#include <avr/io.h>
#include <avr/interrupt.h>

#define F_CPU 16000000UL  // Frecuencia del reloj (16MHz para Arduino)
#define BAUD 9600
#define MYUBRR F_CPU/16/BAUD-1

void USART_Init(unsigned int ubrr) {
    // Configurar el baud rate
    UBRR0H = (unsigned char)(ubrr>>8);
    UBRR0L = (unsigned char)ubrr;

    // Habilitar transmisor
    UCSR0B = (1<<TXEN0);

    // Configurar formato del frame: 8 bits de datos, 1 bit de parada
    UCSR0C = (1<<UCSZ01)|(1<<UCSZ00);
}

void USART_Transmit(unsigned char data) {
    // Esperar hasta que el buffer de transmisión esté vacío
    while (!(UCSR0A & (1<<UDRE0)));

    // Poner el dato en el buffer, envía el dato
    UDR0 = data;
}
```

```c
int main(void) {
    cli(); // Deshabilitar interrupciones

    DDRB = 0x20; // Configura el pin 5 del puerto B como salida (LED)
    PORTD |= 0x04; // Activa la resistencia pull-up en el pin 2 del puerto D
    EICRA = 0b00001000; // Configura INT1 para disparar en flanco de bajada
    EIMSK = 2; // Habilita la interrupción INT1

    // Inicializar UART
    USART_Init(MYUBRR);

    sei(); // Habilitar interrupciones

    while(1) {
        // Bucle infinito vacío
    }
}

ISR(INT1_vect) {
    PINB = 0x20; // Cambia el estado del pin 5 del puerto B (LED)
    USART_Transmit('A'); // Transmite la letra 'A' por UART
    // Puedes cambiar 'A' por cualquier otro carácter ASCII
}
```

Figura 5 Código de transmisión UART

Configuración del Osciloscopio

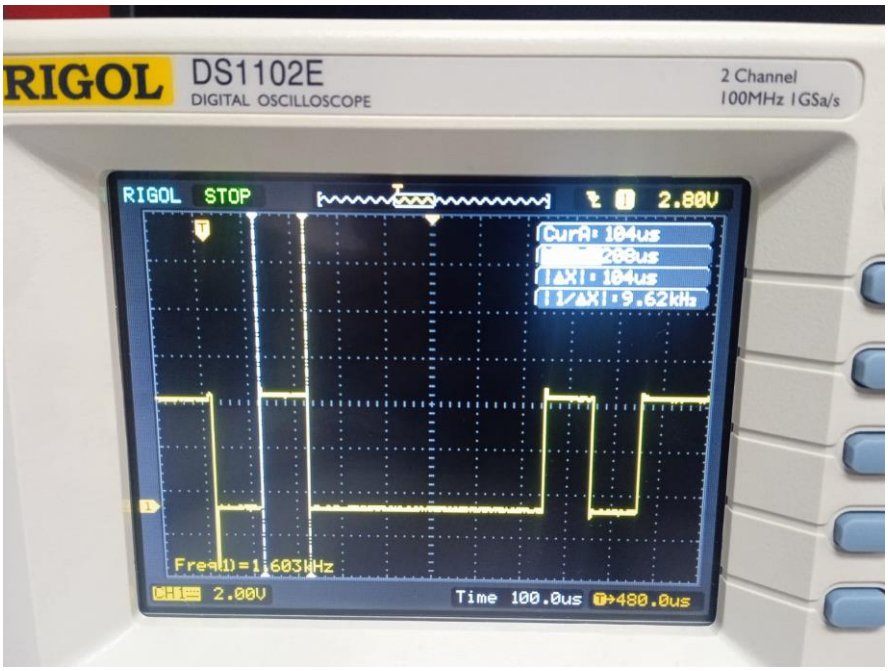
- Conexión: Sonda CH1 en pin TX (D1) del Arduino.
- Escala de tiempo: 100 µs/div (para ver cada bit a 9600 baudios).

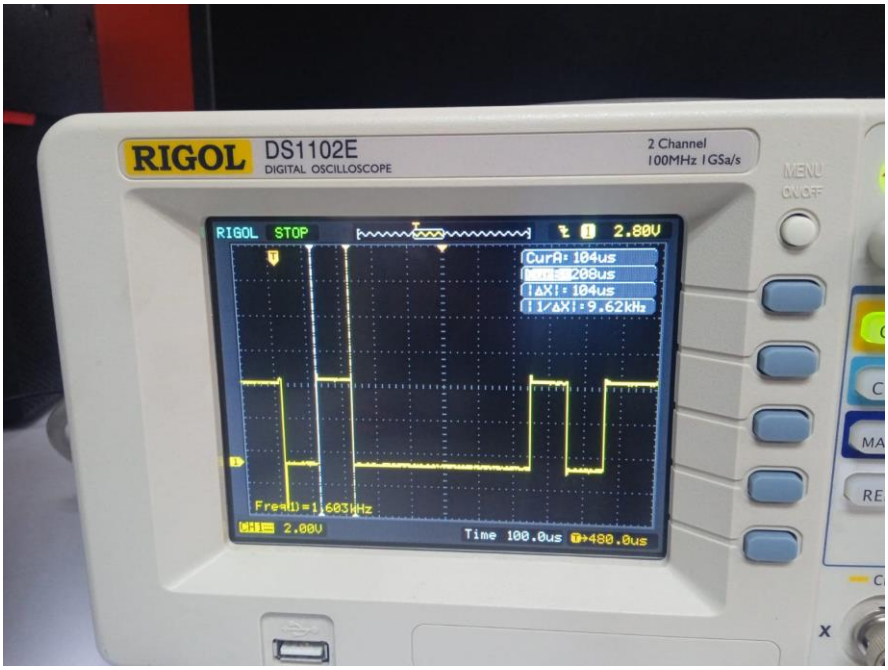
- Trigger: Falling edge en el bit de start.

Análisis de la Señal UART

Para la letra 'A', la trama UART se ve así:

1. Start bit (0) → Nivel bajo.
2. Datos (LSB primero): 1 0 0 0 0 0 1 0 (0x41 en binario).
3. Stop bit (1) → Nivel alto.


RIGOL
DS1102E
DIGITAL OSCILLOSCOPE
2 Channel
100MHz 1GSa/s
RIGOL STOP
2.80V
CurA: 104us
|ΔX|: 104us
|1/ΔX|: 9.62kHz
Freq(1)=1.603kHz
CH1 2.00V
Time 100.0us
480.0us


RIGOL
DS1102E
DIGITAL OSCILLOSCOPE
2 Channel
100MHz 1GSa/s
MENU
ON/OFF
RIGOL STOP
2.80V
CurA: 104us
|ΔX|: 104us
|1/ΔX|: 9.62kHz
Freq(1)=1.603kHz
CH1 2.00V
Time 100.0us
480.0us

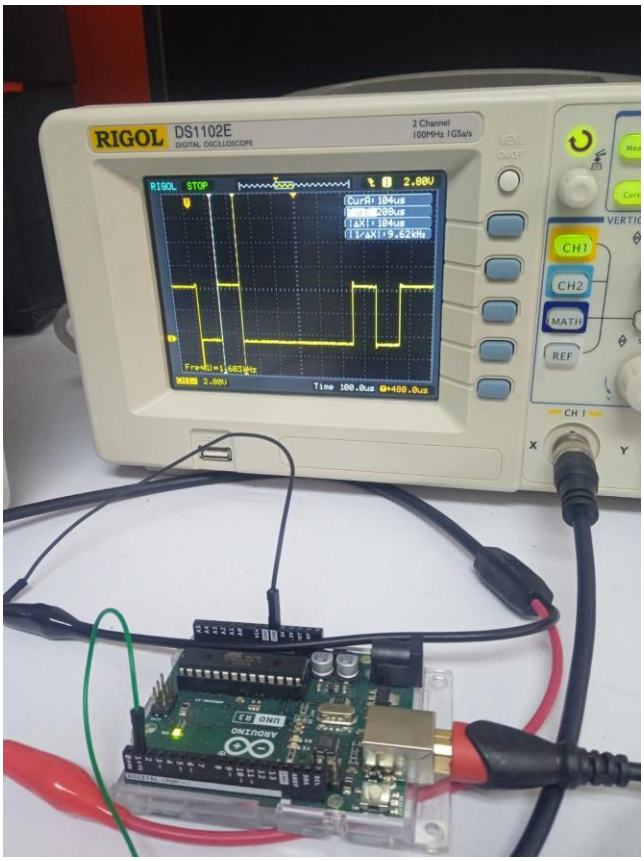

RIGOL
DS1102E
DIGITAL OSCILLOSCOPE
2 Channel
100MHz 1GSa/s
RIGOL STOP
2.80V
CH1
CH2
MATH
REF
ARDUINO
UNO R3

Figura 6

Conclusiones

El uso del osciloscopio Rigol fue esencial para analizar las señales en el dominio del tiempo, permitiendo una comparación entre los valores teóricos y los resultados experimentales.

En la primera actividad, se verificó que el Arduino puede generar una onda cuadrada en uno de sus pines de salida, aunque la frecuencia observada fue un poco menor a la del reloj principal (16 MHz).

En la segunda actividad, se confirmó que el protocolo UART permite transmitir datos de forma asíncrona con una estructura bien definida. Al visualizar la letra 'A' (0x41 en ASCII) en el osciloscopio, se identificaron claramente los bits de start, datos y stop, validando los conceptos teóricos. Sin embargo, se observó que pequeños errores en la configuración

Referencias

[1] D. Features, “8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash,” Microchip.com. [Online]. Available: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf. [Accessed: 25-Mar-2025].

[2] Rohde and Schwarz International, “Qué es UART,” Rohde-schwarz.com. [Online]. Available: https://www.rohde-schwarz.com/es/productos/test-y-medida/essentials-test-equipment/digital-oscilloscopes/que-es-uart_254524.html. [Accessed: 25-Mar-2025].